

Python: module vcs.outfill

vcs.outfill

[index](#)

Outfill (Gfo) module

Modules

[vcs.Canvas](#)
[vcs.VCS validation functions](#)

[vcs. vcs](#)
[cdtime](#)

[vcs.queries](#)
[vcs](#)

Classes

[builtin .object](#)
[Gfo](#)

class **Gfo**([builtin .object](#))

Class: [Gfo](#)

Outfill

Description of [Gfo](#) Class:

The outfill graphics method fills a set of integer values in any Its primary purpose is to display continents by filling their area by a surface type array that indicates land, ocean, and sea-ice example below shows how to apply the outfill graphics method and Fillarea and outfill attributes.

Other Useful Functions:

a=vcs.init()	# Constructor
a.show('outfill')	# Show predefined outfill gr
a.show('line')	# Show predefined VCS line ol
a.setcolormap("AMIP")	# Change the VCS color map
a. <u>outfill</u> (s,o,'default')	# Plot data 's' with outfill 'default' template
a.update()	# Updates the VCS Canvas at
a.mode=1, or 0	# If 1, then automatic update 0, then use update function update the VCS Canvas.

Example of Use:

a=vcs.init()

To Create a new instance of outfill use:

out=a.createoutfill('new','quick')	# Copies content of 'quick'
out=a.createoutfill('new')	# Copies content of 'default'

```

To Modify an existing outfill use:
out=a.getoutfill('AMIP_psl')

out.list()                                     # Will list all the outfill .
out.projection='linear'
lon30={-180:'180W', -150:'150W', 0:'Eq'}
out.xticlabels1=lon30
out.xticlabels2=lon30
out.xticlabels(lon30, lon30)                  # Will set them both
out.xmtics1=''
out.xmtics2=''
out.xmtics(lon30, lon30)                     # Will set them both
out.yticlabels1=lat10
out.yticlabels2=lat10
out.yticlabels(lat10, lat10)                  # Will set them both
out.ymtics1=''
out.ymtics2=''
out.ymtics(lat10, lat10)                     # Will set them both
out.datawc_y1=-90.0
out.datawc_y2=90.0
out.datawc_x1=-180.0
out.datawc_x2=180.0
out.datawc(-90, 90, -180, 180)              # Will set them all
xaxisconvert='linear'
yaxisconvert='linear'
out.xyscale('linear', 'area_wt')             # Will set them both

Specify the outfill fill values:
out.outfill=([0,1,2,3,4])                   # Same as below
out.outfill=(0,1,2,3,4)                      # Will specify the outfill v

There are four possibilities for setting the color index (Ex):
out.fillareacolor=22                         # Same as below
out.fillareacolor=(22)                        # Same as below
out.fillareacolor=([22])                      # Will set the outfill to a
                                                #      color index
out.fillareacolor=None                       # Turns off the color index

```

Methods defined here:

```

__init__(self, parent, Gfo_name=None, Gfo_name_src='default', createGfo=0)

datawc(self, dsp1=1e+20, dsp2=1e+20, dsp3=1e+20, dsp4=1e+20)

list(self)

rename = renameGfo(self, old_name, new_name)
#####
# Function:      renameGfo
#

```

```

# Description of Function:
#      Private function that renames the name of an existing
#      graphics method.
#
#
# Example of Use:
#      renameGfo(old_name, new_name)
#          where: old_name is the current name of outfill
#                  new_name is the new name for the outfill
#
#####
# script(self, script_filename=None, mode=None)
#     Function:      script                                # Calls _vcs.s
#
# Description of Function:
#     Saves out a outfill graphics method in Python or VCS so
#     designated file.
#
# Example of Use:
#     script(scriptfile_name)
#         where: scriptfile_name is the output name of the
#                 mode is either "w" for replace or "a" for
#                 default a Python script will be produced.
#
# Note: If the the filename has a ".py" at the end
#       Python script. If the filename has a ".scr"
#       produce a VCS script. If neither extension
#       default a Python script will be produced.
#
#     a=vcs.init()
#     out=a.createoutfill('temp')
#     out.script('filename.py')                         # Append to a Python fil
#     out.script('filename.scr')                      # Append to a VCS file "
#     out.script('filename', 'w')
#
xmtics(self, xmt1="", xmt2="")
xticlabels(self, xtl1="", xtl2="")
yscale(self, xat="", yat="")
ymtics(self, ymt1="", ymt2="")
yticlabels(self, ytl1="", ytl2="")



---


Properties defined here:

datawc_calendar
    get">get = _getcalendar(self)
    set">set = _setcalendar(self, value)

datawc_timeunits

```

```
get">get = _gettimeunits(self)
set">set = _settimeunits(self, value)

datawc_x1
get">get = _getdatawc_x1(self)
set">set = _setdatawc_x1(self, value)

datawc_x2
get">get = _getdatawc_x2(self)
set">set = _setdatawc_x2(self, value)

datawc_y1
get">get = _getdatawc_y1(self)
set">set = _setdatawc_y1(self, value)

datawc_y2
get">get = _getdatawc_y2(self)
set">set = _setdatawc_y2(self, value)

fillareacolor
get">get = _getfillareacolor(self)
set">set = _setfillareacolor(self, value)

fillareaindex
get">get = _getfillareaindex(self)
set">set = _setfillareaindex(self, value)

fillareastyle
get">get = _getfillareastyle(self)
set">set = _setfillareastyle(self, value)

name
get">get = _getname(self)
set">set = _setname(self, value)

outfill
get">get = _getoutfill(self)
set">set = _setoutfill(self, value)

projection
get">get = _getprojection(self)
set">set = _setprojection(self, value)

xaxisconvert
get">get = _getxaxisconvert(self)
set">set = _setxaxisconvert(self, value)

xmtics1
get">get = _getxmtics1(self)
set">set = _setxmtics1(self, value)

xmtics2
```

```

    get">get = _getxmtics2(self)
    set">set = _setxmtics2(self, value)

xticlabels1
    get">get = _getxticlabels1(self)
    set">set = _setxticlabels1(self, value)

xticlabels2
    get">get = _getxticlabels2(self)
    set">set = _setxticlabels2(self, value)

yaxisconvert
    get">get = _getyaxisconvert(self)
    set">set = _setyaxisconvert(self, value)

ymtics1
    get">get = _getymtics1(self)
    set">set = _setymtics1(self, value)

ymtics2
    get">get = _getymtics2(self)
    set">set = _setymtics2(self, value)

yticlabels1
    get">get = _getyticlabels1(self)
    set">set = _setyticlabels1(self, value)

yticlabels2
    get">get = _getyticlabels2(self)
    set">set = _setyticlabels2(self, value)

```

Data and other attributes defined here:

```

__slots__ = ['setmember', 'parent', 'name', 'g_name', 'xaxisconvert', 'yaxisconvert', 'fillareacolor', 'filloutlinecolor',
'outfill', 'projection', 'xticlabels1', 'xticlabels2', 'yticlabels1', 'yticlabels2', 'xmtics1', 'xmtics2', 'ymtics1',
'datawc_x1', ...]

```

g_name = <member 'g_name' of 'Gfo' objects>

parent = <member 'parent' of 'Gfo' objects>

setmember = <member 'setmember' of 'Gfo' objects>

Functions

```

getGfomember(self, member)
#####
#
# Function:      getGfomember

```

```

#
# Description of Function:
#     Private function that retrieves the outfit members from the
#     structure and passes it back to Python.
#
#
# Example of Use:
#     return_value =
#     getGfomember(self, name)
#         where: self is the class (e.g., Gfo)
#                 name is the name of the member that is being
#
#####
getmember = getGfomember(self, member)
#####
#
# Function:      getGfomember
#
# Description of Function:
#     Private function that retrieves the outfit members from the
#     structure and passes it back to Python.
#
#
# Example of Use:
#     return_value =
#     getGfomember(self, name)
#         where: self is the class (e.g., Gfo)
#                 name is the name of the member that is being
#
#####
renameGfo(self, old_name, new_name)
#####
#
# Function:      renameGfo
#
# Description of Function:
#     Private function that renames the name of an existing outfit
#     graphics method.
#
#
# Example of Use:
#     renameGfo(old_name, new_name)
#         where: old_name is the current name of outfit graphics
#                 new_name is the new name for the outfit graphics
#
#####
setGfomember(self, member, value)
#####
#

```

```

# Function:      setGfomember
#
# Description of Function:
#      Private function to update the VCS canvas plot. If the canvas
#      set to 0, then this function does nothing.
#
#
# Example of Use:
#      setGfomember(self,name,value)
#          where: self is the class (e.g., Gfo)
#                  name is the name of the member that is being updated
#                  value is the new value of the member (or attribute)
#
#####
#setmember = setGfomember(self, member, value)
#####
#
# Function:      setGfomember
#
# Description of Function:
#      Private function to update the VCS canvas plot. If the canvas
#      set to 0, then this function does nothing.
#
#
# Example of Use:
#      setGfomember(self,name,value)
#          where: self is the class (e.g., Gfo)
#                  name is the name of the member that is being updated
#                  value is the new value of the member (or attribute)
#
#####

```

Data

StringTypes = (<type 'str'>, <type 'unicode'>)